DØ Grid Production Architecture: A Higher Level View

Adam Lyon

August 24, 2007

Document CD-2171 “DØ Grid Production Architecture” describes the architecture of the system consisting of the `d0reprotools`, `automc`, `SAMGrid`, and `d0runjob` components along with recommendations on ways to make the system less application specific. This document takes a higher level view of production on the Grid, describing the ideal end-to-end system and giving recommendations on how to take the current system closer to that ideal picture.

1. Introduction

The production system has a seemingly simple task: run a set of executables to transform input files (detector RAW files, Monte Carlo card files) into output files suitable for physics analysis. Of course the details make this process very complicated. The executables themselves have complicated environments and setup requirements. Running the executables on the Grid requires many pieces of software and components. Coordinating the production and handling errors is difficult and time-consuming. In the ideal world, the entire process would be controlled by computer software to greatly reduce the operations load and the chance for mistakes. In the real world, the details of the process make this picture difficult, but not impossible, to achieve.

The ideal DØ Grid Production system is an end-to-end coherent automated system for primary data processing, reprocessing, and Monte Carlo production that efficiently uses dedicated DØ resources as well as other resources opportunistically. Such a system is not the current reality at DØ or, for that matter, at any other experiment. This document will describe the characteristics of the ideal system, give an overview of the current system, and examine possible paths that can bring DØ closer to the ideal.

2. The Ideal System

Figure 1 is a picture of the ideal system for the case of primary processing. The “Master Control” program is responsible for handing the “super” workflow of the processing. In this case, a request arrives to the Master Control with instructions to process raw files of a particular data run. Master Control launches Reco jobs to reconstruct the Raw data into thumbnail files. Once enough thumbnail files have been

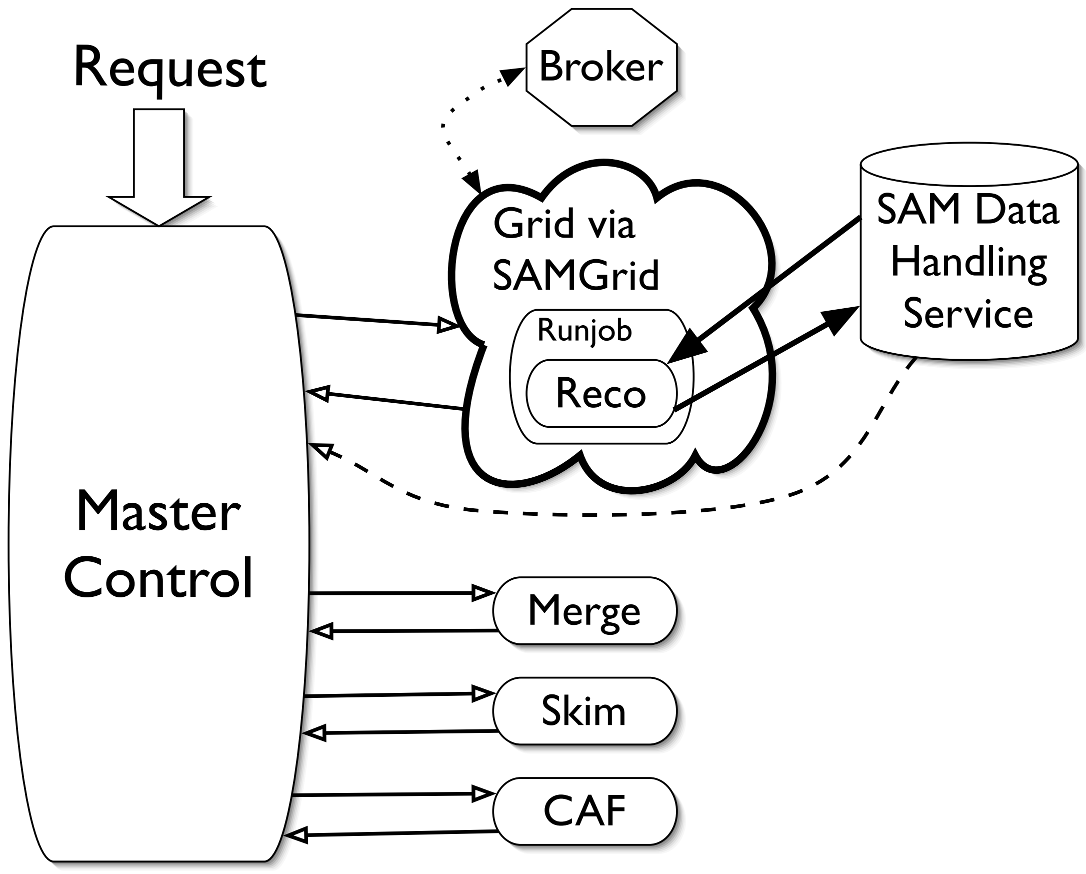

Figure 1: The ideal system for Primary Processing (note that the Reco step is shown in more detail than the others).

created, they are merged into large files suitable for storage. Those merged files are then skimmed to produce many sets of files, each satisfying certain physics requirements. Those files may need to be merged as well (a step not shown in the figure). Finally, the skimmed thumbnails are processed by the CAF maker to produce Root-tuple files.

The Reco step is shown in more detail in the figure. “Launching” the Reco job in the ideal world means submitting the job to computing resources with SAMGrid. SAMGrid can choose an appropriate place to run Reco by contacting a Grid broker or perhaps with specific instructions from Master Control (e.g. always run Reco on the dedicated DØ reconstruction farm within Fermigrid). The SAMGrid system then takes care of running the job, including the specific application workflow embodied in `d0runjob`. Reco and SAMGrid must interact with data handling services to transport executables and data files to the job, as well as transport back output files and/or their meta-data. Before moving to the next step, Master Control would receive a signal back from SAMGrid that the processing was complete, and could also check the data handling catalog for the existence of the new thumbnail files. The other steps are run in a similar manner.

One could imagine running all of the workflow at the application level within d0runjob, and that is how Monte Carlo production is typically run (see Fig. 4). For primary processing and reprocessing though, doing so is somewhat difficult. In order for the merging step to produce large files, files from multiple runs may need to be merged. This may be especially true after skimming. So there is no convenient work package as there is for MC. Furthermore, one may want to run the different steps of the workflow at different places for efficient use of resources.

Master Control would be responsible for error handling and resubmissions for recovery. For example, if a skimming job is submitted to a site never but that job never runs, that fact needs to be determined and acted upon (e.g. resubmitted, perhaps to a different site). Since the data in the output files must correspond to the luminosity recorded by the detector, the integrity of the data must be carefully monitored. Duplicated data and missing data must not occur. These aspects of the error handing are especially difficult to automate.

For this system to work and be maintainable, all of the executables must be compatible with SAMGrid and d0runjob. They should have similar environments and setup procedures (the work to make this happen in SAMGrid and d0runjob are discussed in CD-2171). The executables must be incorporated into the DØ Run Time Environment (RTE) so that DØ libraries are not needed for execution. All RCP files (run control parameter files used to specify the configuration of an executable) must also be in the RTE, which means they must be released in the standard production code (many needed RCP files are in private areas). While none of the requirements listed here are difficult, there is much work to do in order to fulfill them.

3. The current system

The pictures of the current high level system depends on the task to be performed.

3.1. Primary Processing and Reprocessing

High level Master Control for primary processing and reprocessing is currently performed manually by different people. Figure 2 shows the primary processing. Running Reco and merging the output thumbnails fall to Mike Diesburg. Currently, Mike uses the old unsupported SAM v5 farm request system and a set of hand written scripts to run Reco on the dedicated reconstruction farm. The scripts handle processing with Reco as well as merging and storage back into the SAM data handling system. There is some level of error handing, in that if a job on the reco farm fails, it is tried again. The reco executable is portable, in that is uses the DØ Run Time Environment. It can run on systems that do not have the DØ library installed.

In the near future, Reco will run via SAMGrid on the reco farm as a part of Fermigrid with jobs submitted through OSG. Requests will be handled by the new SAM

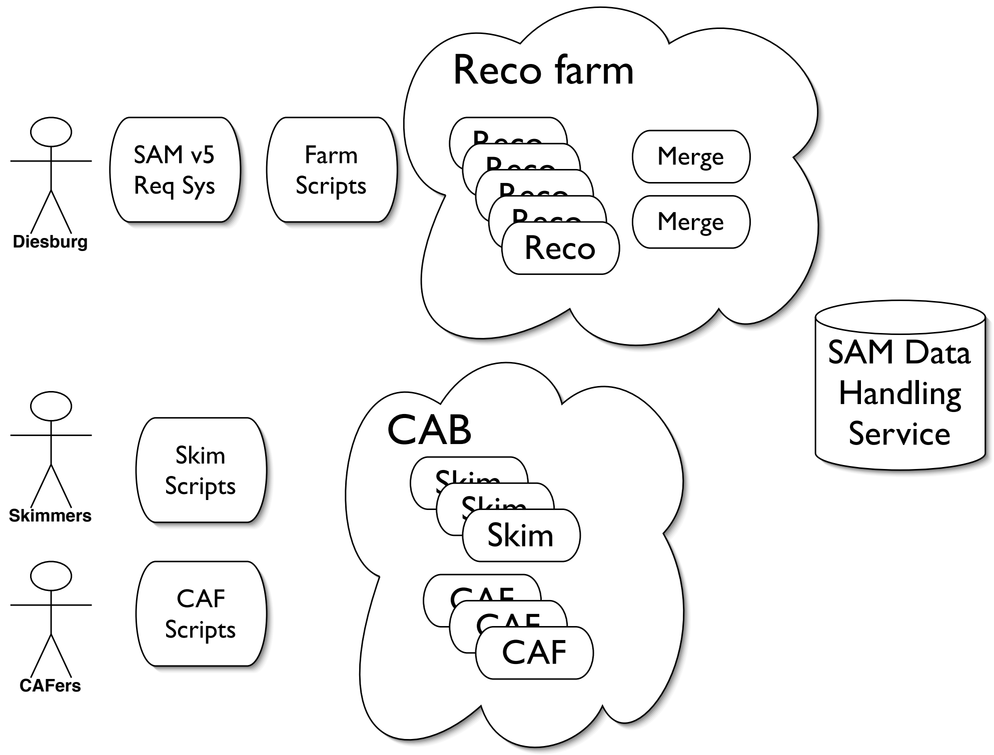

Figure 2: The current Primary Processing system. Note that nearly every executable gets files from and stores files into the SAM Data Handling System.

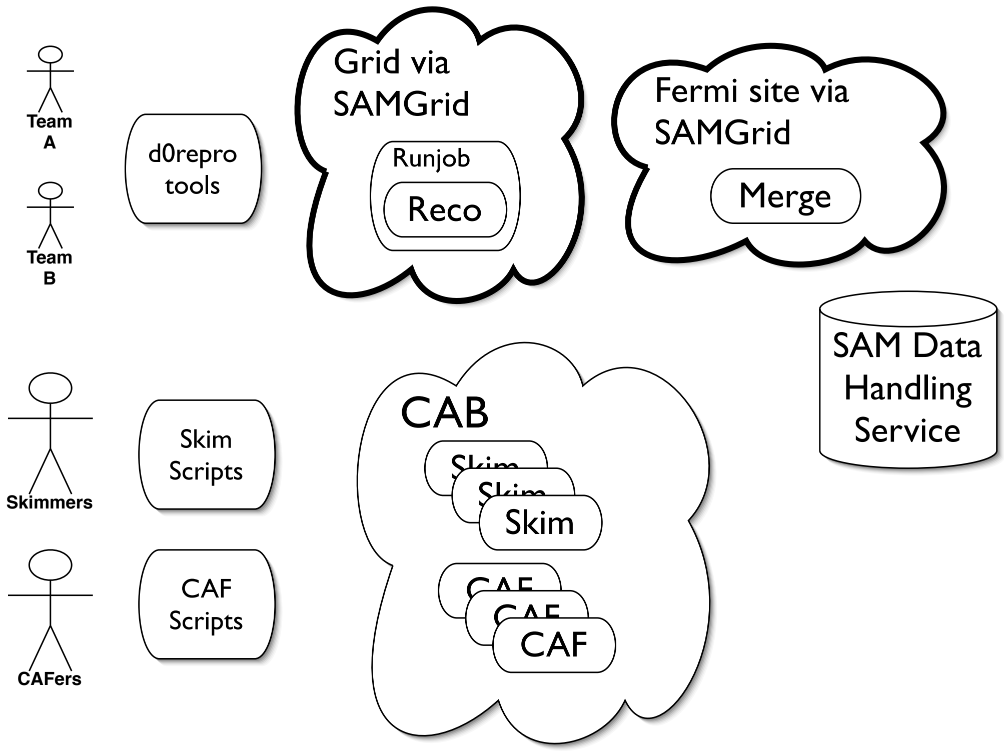

Figure 3: The current Reprocessing system. Note that nearly every executable gets files from and stores files into the SAM Data Handling System.

v7 Request System.

Skimming is handled by another group of people by running the jobs on CAB. They have private hand-written scripts and run the executable out of the DØ library (so the DØ library must be installed on the CAB worker nodes). They use private RCP files (run control parameters) because the default RCP files in the library are not appropriate. Failed jobs must be resubmitted by hand.

CAF Tree production is handled similarly to skimming, but by a different group of people. Again, they use CAB, hand written scripts, and private RCPs.

The latest round of reprocessing used opportunistic resources through SAMGrid as shown in Fig. 3. The scope of the project was to run Reco on raw data files and produce merged thumbnails. Skimming and CAF tree production were handled by the primary processing chain described above. `d0reprotools` at some level served as the Master Control for the reco-merge task in that users could tell it a set of files to

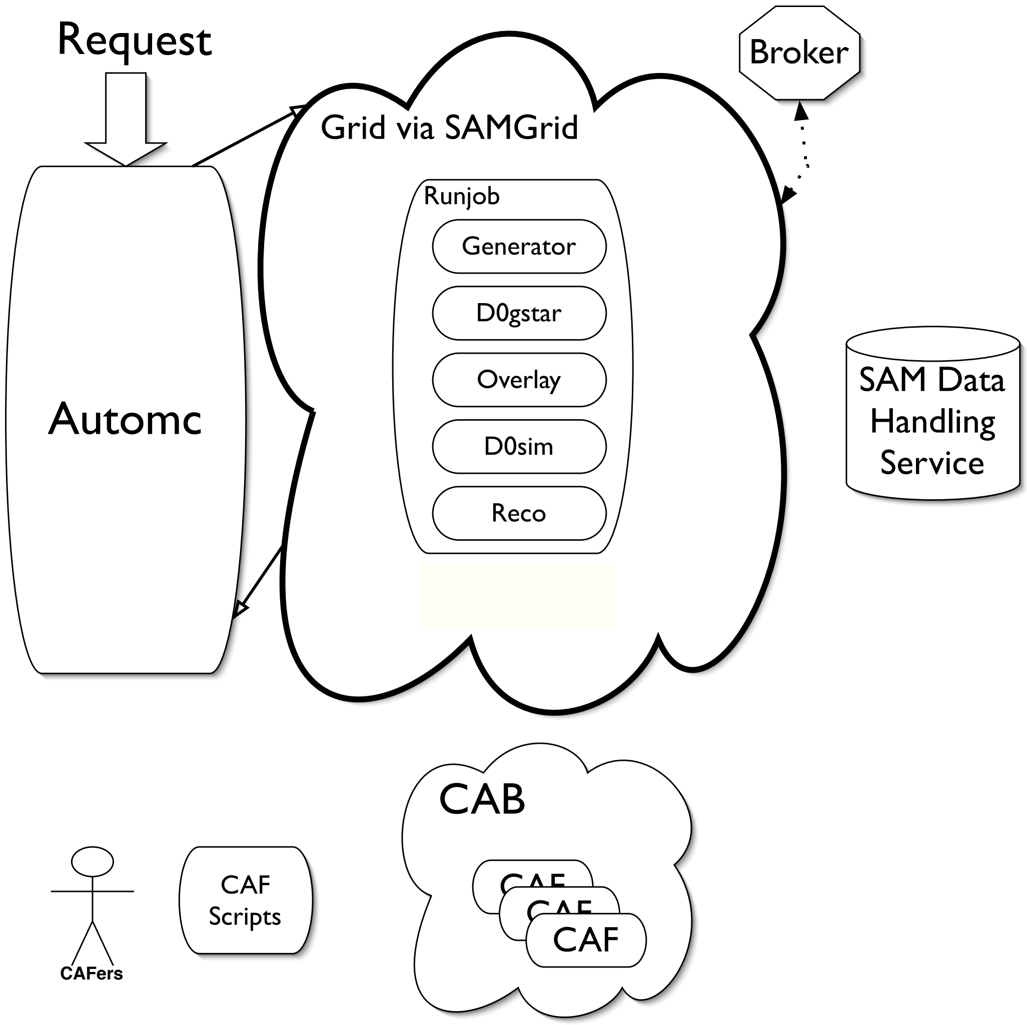

Figure 4: The current Monte Carlo Production system.

process and the program would launch the SAMGrid jobs and also recover from errors by resubmitting (but requiring human intervention to do so). Several teams of people were responsible for feeding jobs into the system. Reco itself was run worldwide on the grid, with merge jobs submitted to the grid but run at Fermilab to optimize network efficiency.

A clear problem with these processes is the lack of coherency. Some tasks are performed on the grid, others on CAB or the reco farm. They use different sets of scripts (some not backed up to CVS) and are maintained by different people. Of course there are reasons for these decisions. The Skim and CAF production code have not been integrated in the standard DØ run time environment (RTE). These jobs require the DØ libraries and so cannot run on the Grid. SAMGrid and Runjob also need development to accommodate these programs. Since the scripts were hand written by the people running them, the maintenance costs and operational load is presumedly low. Transferring these tasks to other people, one overall person, or a Master Control program would be difficult, however.

3.2. Monte Carlo Production

The Monte Carlo Production system shown in Fig. 4 is the closest to the ideal. `Automc` is a continuously running program that waits for new MC production requests from the SAM v7 Request System. It then launches a Grid job via SAMGrid to run a chain of executables to create merged thumbnail files. These files are stored back into SAM where the CAF production group uses their process to create CAF Root-tuple files.

`Automc` needs little human intervention and robustly ensures that the jobs complete on the Grid. It will automatically submit jobs repeatedly if there are errors until they complete successfully.

There is a plan to add the CAF tree maker to the chain of executables run by `d0runjob` so that it would not have to be run separately.

Monte Carlo production nicely lends itself to running the workflow at the application level (in the same job) since an MC request is one neat work-package.

4. An Evolution from Current to Ideal

If the ideal picture is the ultimate goal of DØ, then an evolution is required. How far we go on the path would depend on both human and time resources.

4.1. Phase 1: Ensure current operations

Right now, there is a worry that if a disk dies or if a person has to be unavailable for a period of time, part of the production chain (especially a step in primary production) would stop. In this phase, DØ should…

- Place all scripts and private RCP files into CVS. This step would provide a backup if a private disk with these scripts and files fails.
- Write documentation on how to run the scripts and recover from failed jobs. This step ensures that DØ is not vulnerable to a particular person's absence.
- If possible, replace the incorrect public RCPs with the private RCPs used in production.

4.2. Phase 2: Coherency - Use SAMGrid/RunJob for all executables

An end-to-end system is easier to maintain if all of the components behave the same way. Launching all jobs with SAMGrid and running applications within `d0runjob` would allow for the elimination of the hand-written private scripts. Using SAMGrid would also allow opportunistic use of other resources beyond CAB. In this phase:

- Make SAMGrid more application generic. See CD-2171.
- Do the development work to allow the Skim and CAF Maker executables to run without the need for the DØ libraries to be mounted (use the DØ RTE framework).
- Add the necessary macros to `d0runjob`.
- Prepare standard template JDL files for use with these executables.

Ideally, an automatic submission system would be in place here, but this step is saved for the next phase. In this phase, a very small group of people could more easily run the primary production as all parts of the processing would be submitted the same way. Error recovery would still be difficult and mostly handled by hand, or using d0reprotools for Reco and merging.

At this point improving operations becomes very important in order to maintain a small team size. Failures seem to occur at a much higher rate on the Grid that on CAB. There are many instances of failures of Grid jobs due to problems on specific sites (there is evidence of `automc` resubmitting some jobs over twenty times to a site before they are successfully completed). We would also need robust access to Fermigrid via OSG submission as well. Furthermore, SAMGrid must become much more operations friendly.

In the end, the ease of operations will determine the success of this phase. If operations are too difficult because the failure rate is high, people will return to private scripts to run the jobs on CAB. Bringing SAMGrid to a robust state is crucial, as well as keeping pressure on the Grid system for having up to date and efficient remote sites.

4.3. Phase 3: Automation - Add the Master Control

In this phase, the "super" workflow is automated with the Master Control. Perhaps such a system is an evolution of `automc` or `d0reprotools`. Such automation may allow all processing to be handled by perhaps a single person or a team of two people. The difficult part of this step is the error recovery, especially for skimming since there are multiple output files per input file. If the robustness of the Grid has reached the point where the job failure rate is extremely low, perhaps such errors could be handled by hand.

With fewer people watching the system, active monitoring becomes very important. Passive monitoring where someone needs to watch web pages for signs of problems is clearly not on the path leading to low operational costs. Active monitoring means that an operator is alerted and human intervention is required only if a severe failure occurs. The system should be designed to minimize the number of conditions where human intervention is needed. At the same time, statistics should be gathered so that failure trends can be discovered. An automated recovery system should be able to handle most failures, but if, for example, many jobs are being resubmitted at a particular site, it may be prudent for the operator to see this problem in the statistics and send a trouble ticket to the Grid Operations Center. A system that spends most of the time recovering from failures cannot be efficient.

A completely automated system with high efficiency would be difficult to achieve and the time scale for such development is probably long. But perhaps merging `automc` and `d0reprotools` would be a good start. The robustness of the Grid and SAMGrid are directly linked to the success of such an endeavor.

5. Conclusions

This document describes an ideal Grid Production system for DØ as well as the state of the current system. A series of phases are given in order to bring the current system closer to the ideal. In the end, a cost benefit analyses needs to be performed to determine whether the return of executing such a plan out weighs the costs. Indeed, the current system does work, but is vulnerable to disaster if certain steps are not immediately taken (see Phase 1). The human resources of DØ are decreasing rapidly, including the experiment's expertise in computing from non-Fermilab personnel. Making the system more coherent (Phase 2) should make it easier to operate, so long as the software components and the Grid are robust. Phase 3 (complete automation) could reduce operations down to one or two people. On the other hand, if the efficiency of our software and the Grid is not improved, then the operational load will remain high. DØ and the CD should takes steps now to improve operations for running on the Grid and make the lower level Grid Production system more robust.